Moving Object Detection and Segmentation in Urban Environments from a Moving Platform

Dingfu Zhou, Vincent Frémont, Benjamin Quost, Yuchao Dai, Hongdong Li

PII: S0262-8856(17)30114-2
DOI: doi:10.1016/j.imavis.2017.07.006
Reference: IMAVIS 3633

To appear in: *Image and Vision Computing*

Received date: 5 April 2016
Revised date: 20 March 2017
Accepted date: 21 July 2017

Please cite this article as: Dingfu Zhou, Vincent Frémont, Benjamin Quost, Yuchao Dai, Hongdong Li, Moving Object Detection and Segmentation in Urban Environments from a Moving Platform, *Image and Vision Computing* (2017), doi:10.1016/j.imavis.2017.07.006

Moving Object Detection and Segmentation in Urban Environments from a Moving Platform

Dingfu Zhou[a,b], Vincent Frémont[a], Benjamin Quost[a], Yuchao Dai[b], Hongdong Li[b,c]

[a]*Sorbonne Universités, Université de Technologie de Compiègne, CNRS, UMR 7253, Heudiasyc-CS 60 319, 60 203 Compiègne Cedex, France*
[b]*Research School of Engineering, Australian National University*
[c]*ARC Centre of Excellence for Robotic Vision*

Abstract

This paper proposes an effective approach to detect and segment moving objects from two time-consecutive stereo frames, which leverages the uncertainties in camera motion estimation and in disparity computation. First, the relative camera motion and its uncertainty are computed by tracking and matching sparse features in four images. Then, the motion likelihood at each pixel is estimated by taking into account the ego-motion uncertainty and disparity in computation procedure. Finally, the motion likelihood, color and depth cues are combined in the graph-cut framework for moving object segmentation. The efficiency of the proposed method is evaluated on the KITTI benchmarking datasets, and our experiments show that the proposed approach is robust against both global (camera motion) and local (optical flow) noise. Moreover, the approach is dense as it applies to all pixels in an image, and even partially occluded moving objects can be detected successfully. Without dedicated tracking strategy, our approach achieves high recall and comparable precision on the KITTI benchmarking sequences.

Keywords: Moving Object Detection, Ego-Motion Uncertainty, Motion Segmentation

1. Introduction

Over the past decades, many researchers from different fields such as robotics, automotive engineering and signal processing have been devoting themselves to the development of intelligent vehicle systems. Making the

vehicles to automatically perceive and understand their 3D environment is a challenging and important task. Due to the improvement of the sensor technologies, processing techniques and researchers' contributions, several Advanced Driver Assistance Systems (ADASs) have been developed for various purposes such as forward collision warning systems, parking assist systems, blind spot detection systems and adaptive cruise control systems. Currently, some popular Vision-based Simultaneous Localization and Mapping (VSLAM) and Structure-from-Motion (SfM) [1] systems have also been applied in ADASs or autonomous vehicle, such as the recent popular ORB-SLAM [2].

However, most of these systems assume a static environment, and thus have faced some difficulties in inner urban areas where dynamic objects are frequently encountered. Usually, moving objects are considered as outliers and RANSAC strategy is applied to get rid of them efficiently. However, this strategy will fail when the moving objects are the dominant part of the image. Thus, efficiently and effectively detecting moving objects turns out to be a crucial issue for the accuracy of such systems.

In this article, we focus on the specific problem of moving object detection. We propose a detection and segmentation system based on two time-consecutive stereo images. The key idea is to detect the moving pixels by compensating the image changes caused by the global camera motion. The uncertainty of the camera motion is also considered to obtain reliable detection results. Furthermore, color and depth information is also employed to remove some false detection.

1.1. Related Works

Moving object detection has been investigated for many years. Background subtraction is a commonly used approach for tackling this problem in videos obtained from a static camera: then, regions of interest can easily be detected [3]. Adaptive Gaussian Mixture Models are well known for modelling the background by recursively updating the Gaussian parameters and simultaneously setting the appropriate number of components for each pixel [4]. However, background subtraction cannot be applied to handle the problem when the camera also moves. Due to the camera motion, both the camera and objects motions are coupled in the apparent 2D motion field. The epipolar constraint is classically used for motion detection between two

views [5]. However, it fails in a degenerate case [1]. Other constraints, such as the flow vector bound constraint [6, 7] have been used together with epipolar constraint to detect the degenerate motion.

An alternative to detecting moving objects using the fundamental matrix is 2D planar homography [8, 9]. Homography is used as a global image motion model that makes it possible to compensate the camera motion between two consecutive frames. Pixels which are consistent with the homography matrix are recognized as the static planar background, while these inconsistent ones may belong to moving objects or to static 3D structure with large depth variance (parallax pixels). In order to remove the parallax pixels, additional geometric constraints [8] or clustering strategies [9] may be used.

Compared to monocular vision, stereo vision system (SVS) provides depth or disparity information using images provided by the left and right cameras. Dense or sparse depth/disparity maps computed by global [10] or semi-global [11] matching approaches can be used to build 3D information on the environment. Theoretically, by obtaining the 3D information, any kind of motion can be detected, even the case of degenerate motion mentioned above. In [12], 3D point clouds are reconstructed from linear stereo vision systems first and then objects are detected based on a spectral clustering technique from the 3D points. Common used methods for Moving Object Detection (MOD) in stereo rig can be divided into sparse feature based [13, 14] and dense scene flow-based approaches [15, 16, 17].

Sparse feature-based approaches fail when few features are detected on the moving objects. Then, dense flow-based methods can be used instead. In [15], a prediction of the optical flow between two consecutive frames is calculated based on a function of the current scene depth and ego-motion. From the difference between the predicted and measured flow fields, large non-zero regions are classified as potential moving objects. Although this motion detection scheme provides dense results, the system may be prone to producing a large number of misdetections due to the noise involved in the perception task. Other improved approaches have been developed [18] and [16] to limit misdetections, by considering the uncertainties on the 3D scene flow [18] or on the 2D real optical flow [16]. However, such approaches roughly model the uncertainty of the ego-motion obtained from other sensors

[1] The 3D point moves along the epipolar plane formed by the two camera centers and the point itself, whereas its 2D projections move along the epipolar lines.

(GPS or IMU). In fact, the camera ego-motion has a global influence on the predicted optical flow; therefore, its uncertainty should be well considered to improve detection accuracy.

1.2. Structure and Contributions of This Paper

In this paper, we aim at detecting the foreground moving object from two consecutive stereo image pairs. An object is considered as moving if its location in the absolute world coordinate frame changes between two consecutive frames. This article is an extension of previously-published conference papers [19, 20] with a new review of the relevant state-of-the-art, new theoretical developments and extended experimental results. In [19], we proposed a framework to detect moving objects using the Residual Image Motion Flow (RIMF). In order to improve detection performance, we refined this framework in [20] by considering the uncertainty during the RIMF computation procedure. Both of these works have never been published together; and the aim of this paper is to show the global scope of vision-based perception systems that have been proposed for moving objects detection in intelligent vehicles applications.

The main additional contributions of the present paper are the following: first, we propose a dense moving object detection and segmentation system by using two consecutive stereo frames, which effectiveness is demonstrated on the public KITTI dataset. Unlike in [21] where moving objects are detected by tracking based on sparse features, we compute the dense optical flow in every image pixel. This makes it possible to detect small and partly-occluded moving objects. Next, compared to [20], we add color together with depth information into the graph-cuts framework to improve object segmentation. Then, the 3D density map is used to generate bounding boxes: this strategy proves to be effective to avoid redundancy detection, such as shadow. Finally, we evaluate our proposed moving object segmentation algorithm at the pixel level on the KITTI dataset. At the same time, we also test it at the bounding box level on different real traffic sequences with ground truth. Its effectiveness is demonstrated with respect to related works [15].

This paper is organized as follows: First, Section 2 gives an overview of our proposed moving object detection system. Next, moving pixel detection and motion segmentation are introduced in details in Sections 3 and 4 respectively. Then, we evaluate our proposed system on different KITTI image sequences. The experimental results and analysis are presented in Section 5. Finally, the paper ends with a short conclusion and future works.

2. Problem Setup

As mentioned above, using the epipolar geometry from a monocular camera does not make it possible to detect moving objects when their motion is degenerate. In order to overcome this issue, a stereo system may be employed. The binocular images are recorded by a vehicle-mounted stereo system. We assume that the system is calibrated; thus, a simple rectification [22] can be used to align the left and right images. We denote b as the calibrated baseline for the stereo head. Additionally, the left and right rectified images have identical focal length f and principal point coordinates as $p_0 = (u_0, v_0)^T$.

Given two time-consecutive stereo images from time $t-1$ and t, as shown in Fig. 1, the origin of the world system is assumed to be coincident with the left camera's local coordinate system at time $t-1$. The Z-axis coincides with the left camera optical axis and points forwards, the X-axis points to the right and the Y-axis points downwards. All the coordinate systems are right handed. The main difficulty of moving object detection is caused by the

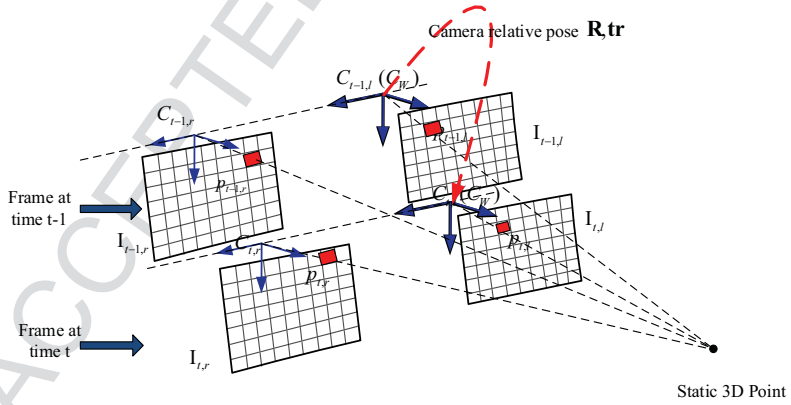


Figure 1: Coordinate frames of the considered stereo-vision system

existence of multiple relative motions – the vehicles' motion and the other objects' independent motions. The position of a pixel extracted from a static background point is $\mathbf{p}_{t-1} = (u_{t-1}, v_{t-1}, 1)^T$ in the previous frame $t-1$, and its image position $\mathbf{p}_t = (u_t, v_t, 1)^T$ in frame t can be predicted by [5, Chapter 9, page 250]:

$$\mathbf{p}_t = \mathbf{K}\mathbf{R}\mathbf{K}^{-1}\mathbf{p}_{t-1} + \frac{\mathbf{K}\mathbf{t}\mathbf{r}}{Z_{t-1}}, \tag{1}$$

where $\mathbf{K}$ is the camera's intrinsic parameter matrix, $\mathbf{R}$, $\mathbf{tr}$ are the relative camera rotation and translation (the pose), and Z_{t-1} stands for the depth of the 3D point X in frame at $t-1$.

In order to detect the moving objects in the image, a straightforward idea is to compensate the camera motion by Eq. (1) first. Then the residual image, calculated as the difference between the current and previous ones compensated in motion, highlights both the pixels belonging to moving objects and the pixels related to motion error estimation. For the sake of clarity, we first define three different flow-based expressions:

- the *Global Image Motion Flow* (GIMF) represents the predicted image changes caused by the camera motion only, that can be calculated using Eq. (1).
- The *Measured Optical Flow* (MOF) represents the real dense optical flow estimated using image processing techniques [23].
- The *Residual Image Motion Flow* (RIMF) is used to measure the difference between MOF and GIMF.

The RIMF can be used to distinguish between pixels related to moving and non-moving objects. In order to calculate the RIMF, the MOF and GIMF should be computed first. Remark that computing the latter requires both information on the camera motion (ego-motion) and on the depth value of the pixels. This paper does not address the issues of computing the dense optical flow [23] and disparity map [24]: we simply use the results from state-of-the-art methods. More precisely, we exploit the approach proposed in [25] in order to compute the dense optical flow and dense disparity map. We then use them directly as inputs of our system. The whole system can be summarized by the following three steps:

1. **Moving pixel detection**. In this step, the moving pixels are detected by compensating the image changes caused by camera motion. In order to improve the detection results, the camera motion uncertainty is considered.
2. **Moving object segmentation**. After the moving pixel detection, a graph-cut based algorithm is used to remove false detections by considering both the color and disparity information.
3. **Bounding box generation**. Finally, the bounding boxes are generated for each moving object by using the UV-disparity map analysis.

For a better understanding, a flowchart of the proposed system is given in Fig. (2), in which the three main steps are highlighted by specific bounding boxes.

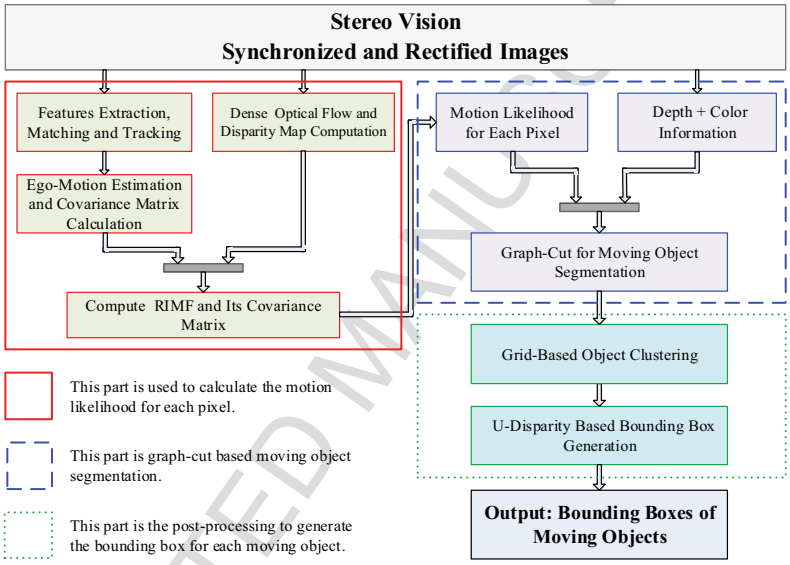


Figure 2: Framework of the moving object detection and segmentation system.

3. Moving Pixel Detection

As described in Fig. 1, four images are considered: two at time $t-1$ and two at time t. The left image $\mathbf{I}_{t-1,l}$ in the previous frame is considered as the reference image. The right image in the previous frame, and the left and right images in the current frame are represented as $\mathbf{I}_{t-1,r}$, $\mathbf{I}_{t,l}$ and $\mathbf{I}_{t,r}$, respectively. Similarly, we define $(u_{t-1,l}, v_{t-1,l})$, $(u_{t-1,r}, v_{t-1,r})$, $(u_{t,l}, v_{t,l})$ and $(u_{t,r}, v_{t,r})$ as corresponding image points in the previous and current stereo frames.

3.1. Ego-Motion Estimation and Uncertainty Computation

Given a set of corresponding points in four images for two consecutive frames, the relative pose of the camera can be estimated by minimizing the sum of the reprojection errors using non-linear minimization approaches. First, the feature points from the previous frame are reconstructed in 3D

via triangulation and using the camera intrinsic parameters. Then these 3D points are re-projected onto the current image frames using the camera motion as below:

$$\hat{\mathbf{x}}_t^i = \mathbf{f}(\mathbf{\Theta}, \mathbf{x}_{t-1}^i) = \begin{bmatrix} Pr^l(\mathbf{K}, \mathbf{\Theta}, \mathbf{x}_{t-1}^i) \\ Pr^r(\mathbf{K}, \mathbf{\Theta}, \mathbf{x}_{t-1}^i) \end{bmatrix}, \tag{2}$$

where $\hat{\mathbf{x}}_t^i = (\hat{u}_{t,l}^i, \hat{v}_{t,l}^i, \hat{u}_{t,r}^i, \hat{v}_{t,r}^i)^T$ are the predicted image points in the current frame and $\mathbf{x}_{t-1}^i = (u_{t-1,l}^i, v_{t-1,l}^i, u_{t-1,r}^i, v_{t-1,r}^i)^T$ are the detected image points in the previous frames. The vector $\mathbf{\Theta} = (r_x, r_y, r_z, tr_z, tr_y, tr_z)^T$ represents the six degrees of freedom of the relative pose. Let Pr^l and Pr^r be the image projections of the 3D world points into the left and right images (non-homogeneous coordinates).

In general, the optimal camera motion vector $\hat{\mathbf{\Theta}}$ can be obtained by minimizing the weighted squared error of measurements and predictions

$$\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{\operatorname{argmin}} F(\mathbf{\Theta}, \mathbf{x}) = \underset{\mathbf{\Theta}}{\operatorname{argmin}} \sum_{i=1}^{N} \|\mathbf{x}_t^i - \mathbf{f}(\mathbf{\Theta}, \mathbf{x}_{t-1}^i)\|_{\Sigma}^2, \forall i = 1 \cdots N. \tag{3}$$

where $\mathbf{x}_t^i = (u_{t,l}^i, v_{t,l}^i, u_{t,r}^i, v_{t,r}^i)^T$ are the matched points in the current frame by using tracking and matching strategies [26] and where $\| \cdot \|_{\Sigma}^2$ stands for the squared Mahalanobis distance according to the covariance matrix Σ.

Although the optimal motion vector $\hat{\mathbf{\Theta}}$ can be obtained by minimizing Eq. (3), its accuracy also depends on the precision of the matched and tracked features' positions in the images. Let $\mathbf{x} = [\mathbf{x}_{t-1}, \mathbf{x}_t] \in \mathbb{R}^{8N}$ represent all points and $\mathbf{x}_{t-1} \in \mathbb{R}^{4N}$,$\mathbf{x}_t \in \mathbb{R}^{4N}$ stand for the points at times $t-1$ and t, respectively. We assume that all points considered in the optimization procedure are well-matched pixel features with only additive Gaussian noise:

$$\mathbf{x} \sim \mathbf{N}(\mu, \mathbf{\Sigma}), \tag{4}$$

where $\mu = (\mu_{\mathbf{x}_{t-1}}, \mu_{\mathbf{x}_t})^T$ and $\mathbf{\Sigma} = \operatorname{diag}(\Sigma_{\mathbf{x}_{t-1}}, \Sigma_{\mathbf{x}_t})$ are the mean and the covariance of the features. The Gauss-Newton optimization of Eq.(3) can converge rapidly if the starting point is close to the optimal point. A real vision-based system requires both a robust estimation of the camera motion and a measurement of the uncertainty associated with this solution. In [27] and [28], the authors proposed a derivation of the covariance matrix using the following model:

$$\Sigma_{\mathbf{\Theta}} = \left(\frac{\partial g}{\partial \mathbf{\Theta}}\right)^{-1} \left(\frac{\partial g}{\partial \mathbf{x}}\right)^{T} \Sigma_{\mathbf{x}} \left(\frac{\partial g}{\partial \mathbf{x}}\right) \left(\frac{\partial g}{\partial \mathbf{\Theta}}\right)^{-T}, \tag{5}$$

where $g(\mathbf{x}, \mathbf{\Theta}) = \frac{\partial F(\mathbf{X}, \mathbf{\Theta})}{\partial \mathbf{\Theta}}$ is the gradient vector of $F(\mathbf{\Theta}, \mathbf{x})$ with respect to $\mathbf{\Theta}$, and where $\Sigma_{\mathbf{x}}$, such as defined in Eq. (4), is the covariance matrix of the measured features at previous and current frames. The partial derivatives, $\frac{\partial g}{\partial \mathbf{\Theta}}$ and $\frac{\partial g}{\partial \mathbf{x}}$ can be computed according to Eq. (5). Alg. (1) presents how the ego-motion and its associated uncertainty can be computed.

Algorithm 1 Ego-motion estimation and error propagation

Require: - Stereo image pairs at previous and current frames;
- Covariance matrix of matched features;

Ensure:
- Relative pose $\mathbf{\Theta}$ ($\mathbf{R}$ and $\mathbf{tr}$) and its covariance $\Sigma_{\mathbf{\Theta}}$;

1: ▶ Features extraction, tracking and matching in four images;
2: ▶ Compute the 3D point at previous frame using Eq. (**??**); ▷ RANSAC process to remove outliers;
3: **for** $i = 1$ **do** N ▷ N is maximum RANSAC times
4: ▶ Randomly select 3 matched features pairs;
5: ▶ iter = 0;
6: **while** iter< 100 || Gauss-Newton increment $> \xi$ **do**
7: ▶ Compute Jacobian matrix and residual matrix;
8: ▶ Update $\mathbf{\Theta}$ using Gaussian-Newton iteration approach ;
9: **end while**
10: ▶ Record $\mathbf{\Theta}$ and inliers indexes if we have more inliers than before;
11: **end for**
12: ▶ Refine the final parameters using all the inliers;
13: ▶ Compute the covariance matrix $\Sigma_{\mathbf{\Theta}}$ using Eq. (5);
14: ▶ return $\mathbf{\Theta}$ and $\Sigma_{\mathbf{\Theta}}$

3.2. Moving Pixel Detection

At the beginning of Section 3, the RIMF has been proposed to detect moving pixels. In order to compute the RIMF, the GIMF should be estimated first. In addition, the uncertainty of RIMF can also be computed from the ego-motion and disparity map uncertainties.

3.2.1. Global Image Motion Flow

The GIMF is used to represent the image motion flow caused by the camera motion. Given a pixel position $\mathbf{p}_{t-1} = (u_{t-1}, v_{t-1}, 1)^T$ in the previous image frame, we can predict its image location $\mathbf{p}_t = (u_t, v_t, 1)^T$ in the

current frame according to Eq. (1). Theoretically, the image location correspondences of a 3D static point in the current frame can be predicted by its depth information in the previous frame and the relative motion information of the camera only. However, this prediction stands only when the 3D point comes from static objects, and it does not hold for dynamic objects. Finally, the GIMF $\mathbf{g} = (g_u, g_v)^T$ for an image point $(u, v)^T$ caused by the camera motion can be expressed as:

$$\mathbf{g} = (g_u, g_v)^T = (u_t - u_{t-1}, v_t - v_{t-1})^T. \tag{6}$$

3.2.2. RIMF Computation

Then, assuming that the MOF estimated between the previous and current frame at point (u, v) is $\mathbf{m} = (m_u, m_v)^T$, the RIMF $\mathbf{q} = (q_u, q_v)^T$ is computed as:

$$\mathbf{q} = \mathbf{g} - \mathbf{m} = (g_u - m_u, g_v - m_v)^T. \tag{7}$$

Ideally, the RIMF should be zero for a static point, while it should be greater than zero for moving points. Simply comparing the RIMF absolute difference to a fixed threshold does not lead to satisfying results to differentiate moving pixels from static ones, because points with different 3D world locations have different image motions. Moreover, the estimated uncertainty, e.g. on camera motion or pixel depth, have a different influence on the image points. Ignoring these uncertainties could lead to a large number of false positive detections. The uncertainty of the RIMF mainly comes from four parts. The first and the most important one is the uncertainty from the camera motion estimation because it has a global influence on each pixel according to Eq. (1). In addition, it affects differently the pixels at different locations. The second influence part is the error of the depth estimation and the third comes from the optical flow estimation process. The last one is the pixel location noise which results directly from the image noise (image rectification, camera intrinsic and extrinsic calibration, digital image quantization, etc).

3.2.3. Motion Likelihood Estimation

As mentioned above, a fixed threshold does not lead to a satisfying solution to detect moving pixels. In order to handle this problem, the uncertainty of RIMF is propagated from the sensors to the final estimation using a first order Gaussian approximation. As in Eq.(7), the RIMF is a function of camera motion $\mathbf{\Theta}$, the pixel location (u, v) at previous frame, the disparity d and the measured optical flow (m_u, m_v). The uncertainty of the measured optical

flow is not considered in this work because it only affects the detection results locally. Based on the forward covariance propagation framework in [5], the RIMF covariance can be calculated by using a first-order approximation as below:

$$\Sigma_{RIMF} = \mathbf{J}\Sigma\mathbf{J}^T, \tag{8}$$

where $\mathbf{J}$ represents the Jacobian matrix with respect to each input variable (e.g. the camera motion $\mathbf{\Theta}$, the pixel position (u, v) and the disparity value d in the previous frame) and $\Sigma = diag\,(\Sigma_{\mathbf{\Theta}}, \Sigma_o)$ is the covariance matrix of all the input variables. The covariance matrix of the camera motion is $\Sigma_{\mathbf{\Theta}}$, and that of the disparity values in the estimation process is $\Sigma_o = diag\,(\ \sigma_u^2,\ \ \sigma_v^2,\ \ \sigma_d^2\ )$, where σ_u and σ_v are the variances that are used to describe the pixel quantization error of the camera and σ_d. In [18], the authors proposed that the uncertainty of the disparity map could also be considered as an approximate standard Gaussian Distribution and its variance can be linearly approximated by:

$$\sigma_d(u, v) = \sigma_0 + \gamma U_d(u, v), \tag{9}$$

where σ_0 and γ are two constant parameters, and where $U_d(u, v)$ is the uncertainty on the disparity value at position (u, v). Here, the matching cost is used as a confidence measure of the disparity value (further details can be found in [29]). Compared to the variance of each parameter in Σ, the covariances between the ego-motion parameters, position and the disparity are negligible and the estimation process is difficult.

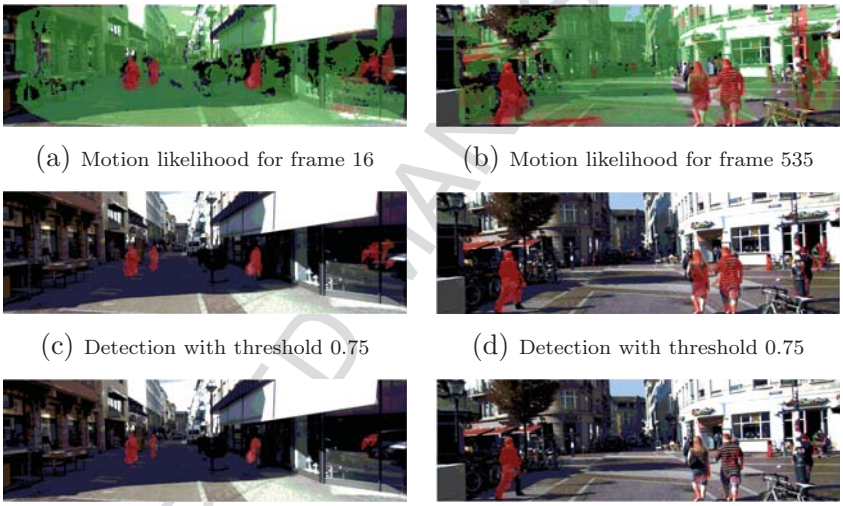
Based on the Σ_{RIMF} estimated above, we can compute the likelihood of a flow vector to be moving. Assuming a stationary world and a Gaussian error propagation, a flow vector is assumed to follow a Gaussian distribution with zero mean and covariance matrix Σ_{RIMF}. Deviations from this assumption can be detected by testing this null hypothesis via a goodness-of-fit. Alternatively, the Mahalanobis distance [30] associated to the RIMF vector can be computed:

$$\mu_{\mathbf{q}} = \sqrt{\mathbf{q}^T\Sigma_{RIMF}^{-1}\mathbf{q}}, \tag{10}$$

where $\mathbf{q}$ is the RIMF vector at a certain image location defined in Eq. (7). Since $\mu_{\mathbf{q}}^2$ is χ^2-distributed, the RIMF motion likelihood $\xi(m)$ of RIMF vector can be computed according to its $\mu_{\mathbf{q}}$ value.

In Fig. (3), the sub-figures (a),(b) are the motion likelihood images resulting from the Mahalanobis distance $\mu_{\mathbf{q}}$. Green pixels are detected as static

and red as moving. In sub-figure 3a, two cyclists come from the opposite direction of the host vehicle and a pedestrian moves in the same direction as the vehicle and all three have been well detected as moving. The shadow of the moving car in the glass window has also been detected. In sub-figure 3b, all the moving pedestrians have been detected, but false positives on the ground are due to MOF errors.



(a) Motion likelihood for frame 16 (b) Motion likelihood for frame 535

(c) Detection with threshold 0.75 (d) Detection with threshold 0.75

(e) Detection with threshold 0.9 (f) Detection with threshold 0.9

Figure 3: Motion likelihood calculation by using MIMF and moving pixel detection using different thresholds.

4. Multi-Cues for Motion Segmentation

A likelihood threshold can be applied to the motion likelihood image so as to distinguish between moving and static pixels. However, detection noise may pervade the process because of the imperfect MOF. Fig. (3) shows some detection results using different thresholds. For example, the motion likelihood estimation at frame 16 (sub-figure 3a) is good and all the moving objects have been well detected, no matter which thresholds are used. Despite that the motion likelihood at frame 535 (sub-figure 3b) is also well estimated, it is still noisy on the edge of static objects due to crude estimates of the optical flow. A lower threshold results in both high true positives and high false positives; conversely, a higher threshold may result in a poor detection rate. An optimal threshold that suits all situations cannot be determined.

4.1. Segmentation Problem

To effectively separate the motion foreground from the background, a segmentation step is adopted here. Usually, the segmentation of image into moving and stationary parts can be considered as a problem of assigning binary labels to each pixel: $l(\mathbf{x}) = 1$ if $\mathbf{x}$ is moving, otherwise $l(\mathbf{x}) = 0$. Several constraints should be considered for segmentation. First, pixels with high motion likelihood should be detected as moving. Second, adjacent pixels with similar appearance and distance should share the same label; otherwise, their labels should be different. By considering all the constraints, an energy function can be built as

$$E(L) = E_r(L) + \lambda E_b(L) \tag{11}$$

where $L = \{l_1, l_2, \cdots, l_p\}$ is a binary vector, p is the number of the pixels in the image, l_i is a binary label for each pixel. Here, E_r and E_b stand for the region and boundary terms and λ is used to balance their influences.

The region term E_r captures the likelihood that the pixels belong to the moving foreground or static background. The motion likelihood of each pixel can be used to build the region term as

$$E_r = -\sum_{\mathbf{x}\in\Omega}\{l(\mathbf{x})\xi_m(\mathbf{x}) + (1 - l(\mathbf{x}))\xi_s(\mathbf{x})\}, \tag{12}$$

where Ω represents the image domain, ξ_m is the motion likelihood and ξ_s is a fixed prior likelihood describing the belief of points being static. Here we assume that all the image pixels share the same stationary likelihood ξ_s since no prior information is available.

The boundary term E_b is used to encourage similar neighboring pixels to be assigned the same label. In order to obtain roust segmentation results, we apply both color [31, 32, 18] and depth information together for building E_b. Since moving objects usually have a significant depth difference with their lateral background, the boundary depth similarity can be defined as:

$$B^d(\mathbf{x}_i, \mathbf{x}_j) = exp(-\sigma(|z(\mathbf{x}_i) - z(\mathbf{x}_j)|) + \alpha), \tag{13}$$

where $z(x_i)$ and $z(x_j)$ represent the depth values at the point x_i and x_j. Note that $B(\ .\ )$ is positive function, monotonically decreasing according to depth difference, in which α and σ are two parameters that control the descent speed and peak value respectively. A bigger value gives a higher

penalization cost to the depth difference, while the value controls the change of cost with the increase of absolute depth difference. Here we empirically set $\alpha = 0$ and $\sigma = \sqrt{2}$ for all experiments. Similarly, the color similarity B^c is also measured as Eq. (13) by considering the color difference. Finally, the boundary term is expressed as

$$E_b = \sum_{\Omega} \sum_{\hat{\mathbf{x}} \in N_4(\mathbf{x})} (B^d(\hat{\mathbf{x}}, \mathbf{x}) + B^c(\hat{\mathbf{x}}, \mathbf{x}))|l(\hat{\mathbf{x}}) - l(\mathbf{x})|, \tag{14}$$

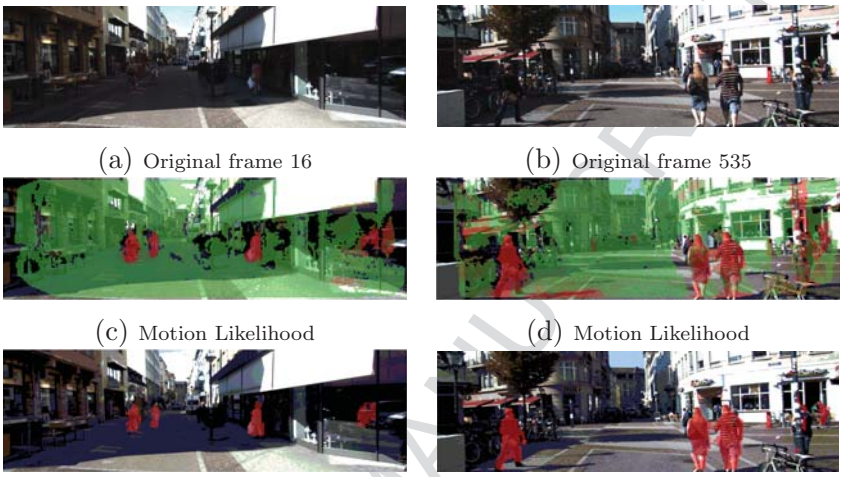
where $N_4(\mathbf{x})$ is the 4-neighborhood of a pixel $\mathbf{x}$.

4.2. Graph-Cut for Motion Segmentation

The minimization of this problem can be solved in the graph-cut framework. A minimum cut, or *min-cut*, is the cut with minimal cost that can be computed using min-cut/max-flow algorithms [33, 31]. In Eq. (11), λ is used to balance the influence between the region and boundary terms. Clearly, the segmentation results heavily depend on the weight parameter λ. For a low value of λ, the segmentation is mainly on the motion likelihood of a single pixel whereas a high value of λ results in only small or no segment at all. In our experiments, we have tested different values: $\lambda \in \{0.25, 0.5, 0.75, 2.0, 5.0\}$. The segmentation results show that small λ result in some error detection, while high λ result in small regions (such as in (d), (e) and (f)). We finally chose $\lambda = 0.5$ for our experiments, which gives good results compared to the other values tested. In order to save computer memory and to improve the processing speed in the graph-cut algorithm, a down-sampling technique is used. We take one pixel out of four in both rows and columns. Fig. (4) displays some of the segmentation results obtained using our approach.

4.3. Bounding Box Generation

A bounding box should be generated around each moving object. Additionally, some erroneously detected pixels (e.g., shadows) should also be eliminated. In our approach, we mainly focused on a cubic detection space of 30 m (longitudinal), 20 m (lateral) and 3 m (height) in front of the vehicle. In this limited subspace, a density map is constructed by projecting all the detected 3D moving points onto the xOz plane. The density map is associated with an accumulation buffer. A cell in the accumulation buffer covers an area of 50 cm $\times$ 50 cm on the xOz plane. The weights that the points



(a) Original frame 16 (b) Original frame 535

(c) Motion Likelihood (d) Motion Likelihood

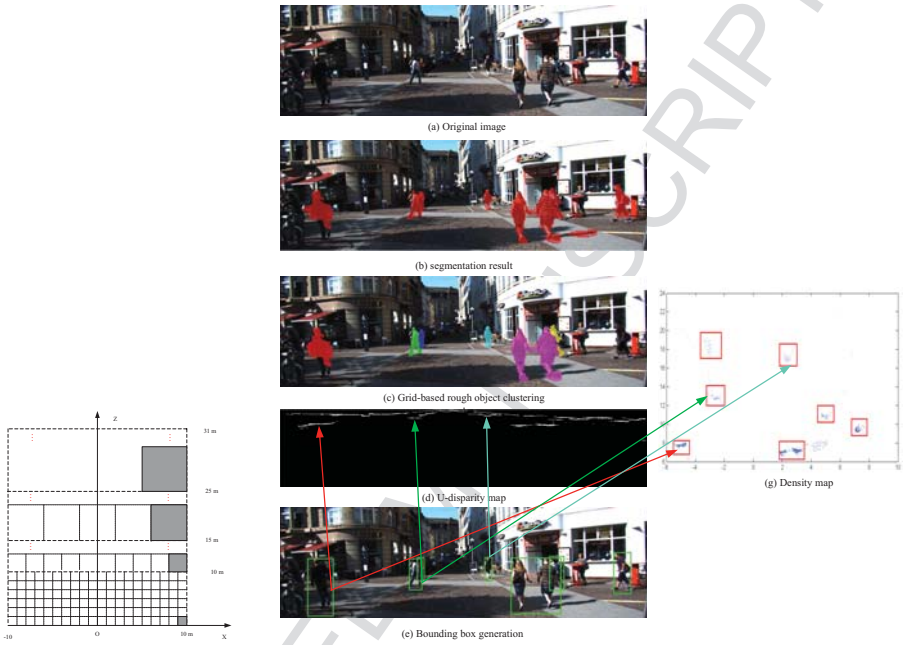
(e) Motion segmentation result (f) Motion segmentation result

Figure 4: Graph-cut based segmentation in different frames.

add to the density map have a Gaussian distribution, with the maximum at the center cell and decreasing in the neighboring cells. Because points become sparser as we move away from the camera, the diameter of the patch increases gradually with the distance. The size of the patch p is defined by the following strategy (as shown in sub-figure 5a):

$$p = \begin{cases} 1 \times 1 \text{ cell} & z < 10\,\text{m} \\ 2 \times 2 \text{ cells} & 10\,\text{m} < z < 15\,\text{m} \\ 4 \times 4 \text{ cells} & 15\,\text{m} < z < 25\,\text{m} \\ 6 \times 6 \text{ cells} & 25\,\text{m} < z < 31\,\text{m} \end{cases} . \tag{15}$$

After obtaining the density map, an empirical threshold is chosen so as to remove sparse points, that could be misdetected image pixels (e.g., shadow or objects borders). Here, a patch will be emptied if its amount of points is below this threshold (e.g. 50). The false alarms at objects boundary are usually due to the error on the measured optical flow (smoothing constraint). Sub-figure 5b shows some ROI generation results relying on the grid-based method. Based on this approach, the shadow can be easily removed, such as in 5b-(c). In 5b-(c), each color corresponds to one rough clustering in the disparity map.



(a) Grid map in the XoZ plane.

(b) Objects clustering.

Figure 5: Bounding boxes generation from moving pixels segmentation.

4.3.1. U-Disparity Map Based ROI Generation

In each cluster, the bounding box can be generated for every moving objects for the next recognition step. Region growing is used to remove redundancies and to integrate part detection using the dense disparity map. U-V disparity maps [34, 35], which are two variants of the classical disparity map, are often used for road and obstacle detection. The U-disparity map has the same width as the original image, which is formed by recording the number of the pixels who share the same disparity value along each image column.

In the U-disparity map, an upright object will form a horizontal line because of similar disparity value. Conversely, each white horizontal line represents a corresponding upright object. This information can be effectively used to determine the width of the objects. After getting the width of the bounding box, region growing [36] is applied to the neighborhood of the clustering group pixels based on the disparity value. The pixels whose

disparity values are between the minimum and maximum disparity value of each cluster are considered to belong to the same object. The final bounding boxes of the moving objects are shown in 5b-(e).

4.3.2. V-Disparity Map-Based Cluster Reduction

According to [37], the real world height of the objects may be estimated by:

$$h_i = h_c + \frac{(y_i - y_0) z \cos\theta}{f} \tag{16}$$

Here, h_i and h_c are respectively the heights of the i^{th} object and of the camera in the world coordinate frame; θ is the camera tilt angle and f is the camera focal length; z is the depth of the object; y_0 and y_i are the horizontal position and top of the object in the image coordinate. Assuming that moving objects are not higher than 3 m, some obvious false positives may be filtered. For this purpose, the horizontal position is first computed using the V-disparity map. Then, the actual height of the objects h_i is calculated using Eq. (16). Finally we retain only the objects whose height is between 0.75 m and 3 m, because the height of most moving objects is in this range. Detailed steps can be found in Alg. (2).

Algorithm 2 Bounding Box Generation and Cluster Reduction

Require: - Objects Bounding box;
- Camera height h_c,camera tilt angle θ and camera focal length f;
- The distance of objects to the camera z;
- Horizon position y_0;

Ensure:
-Real world height of the objects h_i ;

1: ▶ Compute the U- and V- disparity maps;
2: ▶ According to its disparity value, each moving pixel may be assigned to different upright objects using the U-disparity map;
3: ▶ Compute the horizontal line defined by y_0 and the camera tilt θ from the V-disparity map;
4: ▶ Calculate the real world height h_i of the objects using the horizontal line y_0, the camera height h_c and the tilt angle θ as in Eq. (16);
5: ▶ Keep the detected objects for which h_i is between 0.75 m and 3 mm.

5. Experimental Results

Several real image sequences from the KITTI dataset[2] have been chosen to test the effectiveness of our system. More details about the sensor setup and data information can be found in [38, 39]. The actual object labels and locations have been provided in some of these sequences, which can be used to evaluate our moving object detection algorithm. The video sequences are acquired from a SVS installed on the roof of a vehicle. Five different video sequences (fps = 10) in the raw data are chosen for our evaluation. These sequences are captured in the inner city streets, which includes moving vehicles, pedestrians, cyclists etc. In the inner city sequence, the host vehicle was driven at a low speed (about 15 km/h) because of complex road conditions.

Before presenting the experimental results, we review our detection algorithm. First, dense disparity and optical flow [25] are computed before the moving objects detection steps. At the same time, the relative camera pose between two consecutive frames and its covariance are estimated based on sparse feature detection and tracking. The standard deviation of the features in Eq. (4) is empirically set to $\Sigma = diag\,[1.0, 1.0]$ pixel. Ideally this value should be changed depending on the situation. In order to compute the variance of disparity in Eq. (9), we empirically set $\sigma_0 = 0.25$ and $\gamma = 0.075$.

5.1. Quantitative Evaluation

Due to the difficulty of finding the moving object detection and segmentation benchmark in the real traffic scene, we try to construct out own ground truth based on the existing KITTI dataset.

5.1.1. Moving Object Segmentation Evaluation at Pixel Level

In the "Scene Flow Evaluation 2015 benchmark", the ground truth of some moving objects (pixel level) have been provided for the training images. We can use this dataset to evaluate our depth-aided moving object segmentation approach at the pixel level. The precision (P), recall (R) and F-measure (F) are usually computed to measure the performance of the system; they are defined as

$$R = \frac{tp}{tp + fn},\ P = \frac{tp}{tp + fp} \text{ and } F = \frac{2R * P}{R + P}. \tag{17}$$

[2] http://www.cvlibs.net/datasets/kitti/

The true positive (tp) amount represents the number of pixels that have been correctly detected as moving. False positives (fp) are static pixels that have been mis-detected as moving. False negatives (fn) are the moving pixels that have not been detected. The training dataset consists of 200 image groups and each group includes four images: two stereo image pairs at current and next time instant. However, some dynamic objects have been labeled in this training dataset, such as trucks, pedestrians and cyclists. Finally, only 164 image groups, where the moving objects have been fully labeled, have been used for our evaluation.

Some examples of the moving object segmentation results are displayed in Fig. 6. Table 1 displays the moving detection results with or without segmentation. The results show that it is hard to determine an optimal motion likelihood threshold with reasonable precision and recall. A lower threshold gives a high recall value while many false positives have been generated. Inversely, a high threshold will reduce the recall value. By using segmentation, the detection accuracy can be increased: the recall, precision and the F-measure are significantly improved compared to when using a fixed threshold. In addition, compared to [20], the segmentation results have been also improved, likely by adding the color information into the graph-cut framework.

Methods	Recall (R)	Precise (P)	F-measure (F)
Fixed threshold (0.5)	0.7568	0.4993	0.6016
Fixed threshold (0.7)	0.7338	0.6073	0.6646
Fixed threshold (0.9)	0.6315	0.7007	0.6643
Graph-cut with depth	0.7274	0.6997	0.7133
Graph-cut with depth + color	0.7641	0.6959	0.7284

Table 1: Moving objects segmentation evaluation on the KITTI dataset. Different thresholds have been chosen for evaluation, while we set ξ_s= 0.65 for the whole evaluation.

5.1.2. Moving Objects Detection Evaluation at Bounding Boxes Level

In the "Raw dataset" category, 2D bounding boxes of the moving objects have been provided for several sequences with tracklets. We also used these sequences to evaluate our system at the bounding box level. The ground truth of the moving objects are generated by labeling them manually from

Figure 6: Moving object segmentation on the KITTI "Scene Flow Evaluation 2015 benchmark". Sub-figures (a),(b) and (c) display three segmentation examples, the detected moving pixels are drawn in red. In (a), the moving car has been well detected. In (b), the black one right in front of the camera has not been detected because the car locates nearly at the point of epipole. In (c), some background pixels have been detected as moving due to the occlusion in the next frame. In (d), the silvery car has not been detected because it is out of the detection range.

the tracklets for each frame. Here, only the moving objects whose distance is less than 30m are considered. We employ the PASCAL challenge [40] measure to evaluate the detection results:

$$score = \frac{\text{area}(BB_g\{i\} \cap BB_d\{j\})}{\text{area}(BB_g\{i\} \cup BB_d\{j\})}, \tag{18}$$

where BB_d and BB_g are the detected and ground truth bounding boxes of the objects. An object is considered to be correctly detected only when $BB_d\{i\}$ and $BB_g\{j\}$ share a sufficient overlap area. A threshold $score$ is set to determine this overlap area: we chose $score = 0.5$ as in the PASCAL challenge [40]. The precision, recall and F-measure are also computed to measure the performance of the system. In this case, tp represents the number of real moving objects bounding boxes have been correctly detected in the whole sequence, fp stands for static objects that have been misdetected as moving, and fn are the moving objects that have not been detected. The true static objects are not taken into account because our algorithm focuses on detecting moving objects only. Finally, 6 typical image sequences in the "City" category of the "Raw dataset" are taken for our evaluation. Some detection results of these sequences are shown in Fig. 7.

Fig. 7-(a) shows the detection results in sequence 5. This sequence is captured around the corner of a quiet city street. The moving van and

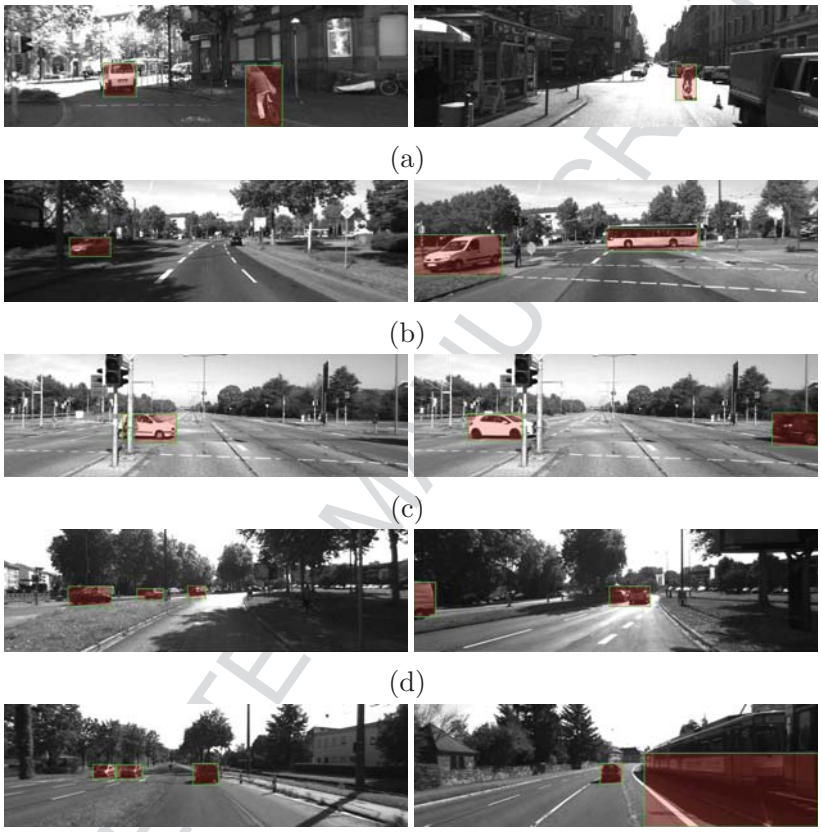


Figure 7: Moving object detection on different sequences. Sub-figures (a), (b), (c), (d) and (e) display the detection results, respectively on sequences 05, 11, 17, 51 and 56 in the category of "City" in the raw data.

the cyclist have been detected in nearly all the frames. The good detection performance benefits from the low camera speed and the relative simple street environment. The van in the right image has not been detected due to the far distant. Fig. 7-(b) and (c) give results of sequences 11 and 17 respectively.

In order to highlight the advantage of our proposal to consider the camera pose and disparity uncertainty, we take the method presented in [15] (which does not consider these uncertainties) as the baseline. In contrast with our proposed method, the RIMF is directly used to detect moving objects. We transform the RIMF into motion likelihood $\mu_{\mathbf{q}}$ as $\mu_{\mathbf{q}} = 1-\exp(-|\mathbf{q}|)$ and take this value as the input of the segmentation step. The motion likelihood in-

creases with the increasing of the RIMF. In order to achieve fair comparison, we keep all the other steps and parameters as the same as in the proposed approach.

Method	Evaluation	05	11	17	18	51	56
Method [15]	Recall	0.781	0.895	0.961	0.910	0.967	0.949
	Precision	0.383	0.675	0.961	0.843	0.556	0.510
	F-measure	0.513	0.770	0.961	0.876	0.706	0.664
Our method	Recall	0.898	0.919	1.00	0.933	0.968	0.787
	Precision	0.690	0.696	1.00	0.849	0.680	0.768
	F-measure	0.780	0.792	1.00	0.890	0.799	0.777

Table 2: Moving object detection evaluation on different "Raw" data sequences.

Tab. 2 illustrates the quantitative evaluation of the two approaches with the ground truth on six different image sequences. For a clear comparison, we have highlighted the best results in blue for each sequence. From the table, we can see that the detection results have been greatly improved for all the sequences. Taking the uncertainties into account improves the detection rate and reduces the false alarm rate.

5.2. Detection Results on KITTI Sequences

Besides the evaluation results mentioned above, we also tested our system on other scenarios in the KITTI dataset. Fig. 8a shows the detection results in the campus sequence (Campus sequence 37). During this sequence, the camera turned from left to right at a high speed: the vehicle direction changes nearly of 90 degrees in 4.3 seconds. The experimental results show that our algorithm can work well in this situation. The cyclists behind the trees far from the camera can be detected. In Fig. 8a, the red rectangle highlights the undetected moving objects. The cyclist has not been detected by our algorithm because it does not appear in the right camera and the 3D points cannot be reconstructed in the disparity map. Two pedestrians at the left boundary of the second image have also been included in one rectangle because they are not separable in the disparity space.

We also tested our algorithm on a suburban highway sequence (Road sequence 16) and the detection results are displayed in Fig. 8b. On the highway, both the ego-vehicle and the other vehicles move at a high speed, about $60\,\mathrm{km\,h^{-1}}$. The frame rate of image sequence is 10 frames per second.

(a) Results on a campus sequence. (b) Results on a suburban road.

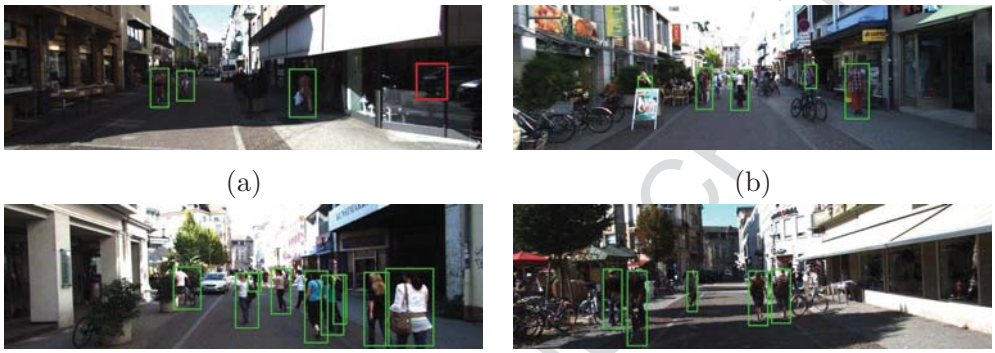
Figure 8: Detection results on KITTI dataset.

In this case, the dense optical flow approach does not work well because of the high changes between two successive frames. Then, sparse feature tracking and matching between two stereo frames can be used for detecting moving objects. A lower threshold is set in the feature extraction step to make sure that we can obtain enough features on the moving objects. The driving vehicles coming from the opposite direction were detected at a range of 40 m, which remains sufficient for an appropriate reaction of the driver. The white car moving in front of the camera was also properly detected even as it moves in the same direction as the ego vehicle.

An interesting thing is that the proposed method can also detect forward moving objects even if it stands in the center of Field-Of-View and has exactly the same speed with the ego-vehicle, because the MOF is zero in this case, while the GIMF caused by camera motion is not. Therefore, the forward vehicle can be detected because the final RIMF is not zero. On the contrary, the proposed method can also consider the distant background existing in the center of FOV as static object because both the MOF and GIMF will be zero in this case.

The last sequence we tested is taken in a crowded street[3]. The host vehicle

[3]A video of the detection results can be found at



(a) (b)

(c) (d)

Figure 9: Detection results on a crowd inner city street.

moves slowly, which makes detecting moving objects easier. Slowly moving objects can well be detected by our approach, even when they move on the epipolar plane. Note that the algorithm also detected partially occluded objects because dense disparity and optical flow maps are used. Some false negative and false positive detection happen in the real image sequences, as displayed in Fig. 9, due to reflections on windows in the scene.

5.3. Computational Time

All the experiments have been realized on a standard laptop (Intel i7, 4 Core) with the Matlab R2015a processing environment. When the dense flow is used, the total average computational time is about 165 seconds for each frame. The dense optical flow and disparity map calculation step takes about 150 seconds. Around 10 seconds are spent on the motion likelihood computation, 4 seconds on the graph-cut based segmentation and 1 second on the bounding boxes generation. Computing ego-motion and estimating the uncertainty only takes about 0.25 seconds. Although our Matlab implementation is not real-time, it is also faster when compared to [41] (7 minutes per frame) and further accelerations could be achieved by C/C++ implementation with parallel/GPU computing.

https://www.youtube.com/watch?v=mfSJnCoyLxc.

6. Conclusion and Future Works

In this paper, an approach has been proposed to detect moving objects from two consecutive stereo frames. The ego-motion uncertainty is estimated through a first-order error propagation model that is used to obtain the motion likelihood for each pixel. Pixels with a high motion likelihood and a similar depth are detected as moving based on a graph-cut motion segmentation approach. Additionally, a fast recognition of moving objects becomes possible based on the segmentation results. Detection results in several different real video sequences show that our proposed algorithm is robust with respect to global (camera motion) and local (optical flow) noise. Furthermore, our approach works with all image pixels and arbitrarily moving objects (including partially occluded) can be detected. Without any tracking strategies, our detection approach gives a high recall rate and also exhibits an acceptable precision rate in several public sequences.

However, the computational complexity of the proposed method is an important concern. This is mainly due to computation of the motion likelihood for every image pixel and the segmentation using the graph-cut algorithm. GPU-based algorithms could be used to overcome this weakness [42]. In addition, the performance of MOD highly relies on the results of dense optical flow and disparity maps. However, their estimation in a complex dynamic environment (including other moving objects) often becomes very difficult.

Acknowledgments

This work was carried out within the framework of the Equipex ROBOTEX (ANR-10- EQPX-44-01). Dingfu Zhou was sponsored by the China Scholarship Council for 3.5 year's PhD study at HEUDIASYC laboratory in University of Technology of Compiegne.

References

[1] S. Song, M. Chandraker, and C. Guest, "High accuracy monocular sfm and scale correction for autonomous driving," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 38, no. 4, pp. 730–743, 2016.

[2] R. Mur-Artal, J. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *Robotics, IEEE Transactions on*, vol. 31, no. 5, pp. 1147–1163, 2015.

[3] S. Y. Elhabian, K. M. El-Sayed, and S. H. Ahmed, "Moving object detection in spatial domain using background removal techniques-state-of-art," *Recent patents on computer science*, vol. 1, no. 1, pp. 32–54, 2008.

[4] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Pattern Recognition, Proceedings of the 17th International Conference on*, vol. 2, pp. 28–31, IEEE, 2004.

[5] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[6] A. Kundu, K. M. Krishna, and J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," in *Intelligent Robots and Systems, International Conference on*, pp. 4306–4312, 2009.

[7] A. Kundu, K. M. Krishna, and C. Jawahar, "Realtime multibody visual slam with a smoothly moving monocular camera," in *Computer Vision (ICCV), IEEE International Conference on*, pp. 2080–2087, IEEE, 2011.

[8] M. Irani and P. Anandan, "A unified approach to moving object detection in 2d and 3d scenes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 6, pp. 577–589, 1998.

[9] A. Bugeau and P. Perez, "Detection and segmentation of moving objects in complex scenes," *Computer Vision and Image Understanding*, vol. 113, no. 4, pp. 459–476, 2009.

[10] L. Wang and R. Yang, "Global stereo matching leveraged by sparse ground control points," in *Computer Vision and Pattern Recognition (CVPR), Conference on*, pp. 3033–3040, IEEE, 2011.

[11] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 2, pp. 807–814, 2005.

[12] S. Moqqaddem, Y. Ruichek, R. Touahni, and A. Sbihi, "Objects detection and tracking using points cloud reconstructed from linear stereo vision," *Current Advancements in Stereo Vision*, p. 161, 2012.

[13] B. Kitt, B. Ranft, and H. Lategahn, "Detection and tracking of independently moving objects in urban environments," in *Intelligent Transportation Systems, 13th International IEEE Conference on*, pp. 1396–1401, IEEE, 2010.

[14] P. Lenz, J. Ziegler, A. Geiger, and M. Roser, "Sparse scene flow segmentation for moving object detection in urban environments," in *Intelligent Vehicles Symposium (IV),IEEE*, pp. 926–932, 2011.

[15] A. Talukder and L. Matthies, "Real-time detection of moving objects from moving vehicles using dense stereo and optical flow," in *Intelligent Robots and Systems, Proceedings. International Conference on*, vol. 4, pp. 3718–3725, IEEE, 2004.

[16] V. Romero-Cano and J. I. Nieto, "Stereo-based motion detection and tracking from a moving platform," in *Intelligent Vehicles Symposium, IEEE*, pp. 499–504, IEEE, 2013.

[17] C. Rabe, T. Müller, A. Wedel, and U. Franke, "Dense, robust, and accurate motion field estimation from stereo image sequences in real-time," in *European conference on computer vision*, pp. 582–595, Springer, 2010.

[18] A. Wedel and D. Cremers, *Stereo scene flow for 3D motion analysis*. Springer Verlag London Limited, 2011.

[19] D. Zhou, V. Frémont, and B. Quost, "Moving objects detection and credal boosting based recognition in urban environments," in *Cybernetics and Intelligent Systems (CIS), Conference on*, pp. 24–29, IEEE, 2013.

[20] D. Zhou, V. Fremont, B. Quost, and B. Wang, "On modeling ego-motion uncertainty for moving object detection from a mobile platform," in *Intelligent Vehicles Symposium Proceedings*, pp. 1332–1338, IEEE, 2014.

[21] P. Lenz, J. Ziegler, A. Geiger, and M. Roser, "Sparse scene flow segmentation for moving object detection in urban environments," in *Intelligent Vehicles Symposium (IV)*, pp. 926–932, IEEE, 2011.

[22] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Inc., 2008.

[23] C. Liu, *Beyond pixels: exploring new representations and applications for motion analysis*. PhD thesis, Massachusetts Institute of Technology, 2009.

[24] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Asian Conference on Computer Vision*, pp. 25–38, Springer, 2010.

[25] C. Vogel, K. Schindler, and S. Roth, "3d scene flow estimation with a piecewise rigid scene model," *International Journal of Computer Vision*, vol. 115, no. 1, pp. 1–28, 2015.

[26] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *Intelligent Vehicles Symposium, IEEE*, pp. 963–968, 2011.

[27] R. M. Haralick, "Propagating covariance in computer vision," in *Performance Characterization in Computer Vision*, pp. 95–114, Springer, 2000.

[28] J. C. Clarke, "Modelling uncertainty: A primer," *University of Oxford. Dept. Engineering science, Tech. Rep*, vol. 2161, p. 98, 1998.

[29] X. Hu and P. Mordohai, "A quantitative evaluation of confidence measures for stereo vision," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 11, pp. 2121–2133, 2012.

[30] P. Mahalanobis, "On the generalised distance in statistics," in *Proceedings of the National Institute of Science of India*, pp. 49–55, 1936.

[31] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in ND images," in *Computer Vision. IEEE International Conference on*, vol. 1, pp. 105–112, 2001.

[32] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," in *ACM Transactions on Graphics (TOG)*, vol. 23, pp. 309–314, 2004.

[33] D. Greig, B. Porteous, and A. H. Seheult, "Exact maximum a posteriori estimation for binary images," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 271–279, 1989.

[34] R. Labayrade, D. Aubert, and J.-P. Tarel, "Real time obstacle detection in stereovision on non flat road geometry through" v-disparity" representation," in *Intelligent Vehicle Symposium*, vol. 2, pp. 646–651, IEEE, 2002.

[35] Z. Hu and K. Uchimura, "UV-disparity: an efficient algorithm for stereovision based scene analysis," in *Intelligent Vehicles Symposium.*, pp. 48–54, 2005.

[36] R. Adams and L. Bischof, "Seeded region growing," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 6, pp. 641–647, 1994.

[37] D. Hoiem, A. A. Efros, and M. Hebert, "Putting objects in perspective," *International Journal of Computer Vision*, vol. 80, no. 1, pp. 3–15, 2008.

[38] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition, Conference on*, pp. 3354–3361, IEEE, 2012.

[39] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[40] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[41] R. K. Namdev, A. Kundu, K. M. Krishna, and C. Jawahar, "Motion segmentation of multiple objects from a freely moving monocular camera," in *Robotics and Automation, International Conference on*, pp. 4092–4099, IEEE, 2012.

[42] Z. Yang, Y. Zhu, and Y. Pu, "Parallel image processing based on cuda," in *Computer Science and Software Engineering, International Conference on*, vol. 3, pp. 198–201, IEEE, 2008.

Highlights

- Dense moving object detection and segmentation by stereo cameras.
- Ego-motion and depth uncertainties are considered for motion detection.
- Color and depth are combined together for improving motion segmentation.